In the Specification:

Please replace the title with the following title:

METHOD FOR PRESENTING A NATURAL LANGUAGE COMPARISON OF ITEMS

Please add the following paragraph at the top of page 1:

This is a continuation-in-part of US Provisional Patent Application No. 60/196,303, filed April 12, 2000.

Please replace paragraph 0049 with the following amended paragraph:

Two major units used to construct the output are a routine named **"cmp_SubstVars"** and another routine named **"update_compare_text"**. **"Cmp_SubstVars"** receives a string where various variables are present and replaces the variables with the relevant content. Whereas the content is to be prepared in advance by the calling routine, **cmp_SubstVars** identifies the variables in the input string provided that the variables are capitalized and enclosed in brackets followed by the % sign, as in: <%VARIABLE_NAME%>. The replacement is done using a simple search of the substrings "<%" and "%>" and looking up the variable enclosed in a look up table. The table contains the names of the accepted variables and pointers to the memory where the content to substitute said variables resides.

Please replace paragraph 0054 with the following amended paragraph:

The first thing **"construct_compare_text"** does is reading the library file described. After this data has been loaded, **"construct_compare_text"** will start assembling the natural language output. The beginning of the output is the similar products

information (located either in "prd_similar_by" entries or in "prd_rel_info" entries if only relational compare has been used). The first entry of this type is fetched and the following operations are performed: first, **"construct_compare_text"** will build a line representing the similarity between products. For this purpose **"construct_compare_text"** will launch a routine named **"construct_similar_products_text"**. This routine, given the names of the similar products and the topic-value pairs by which said products are similar will convert the list of the products to a string by separating the list of products to one product plus the remainder of the list, concatenating each product name in the remainder of the list with "," and concatenating the result with the next product name subsequent**[[ial]]**ly until there are no more elements in the list. The outcome string is then appended to the first product name by using the following formula: "ResultString and ProductName". Of course, when there's only one product the result would be only the name of the product and **"construct_similar_products_text"** will write the flag "_singl" to the memory to state we're discussing a single product. In the above case, where multiple products are discussed, **"construct_similar_products_text"** will write the flag "_mult".